USING THE LTSS-DDA PROVIDER UPLOAD API

Version History

Version	Date
Draft 1.0	1/4/2019
Final	10/25/2019

Overview

The provider upload API is the primary way for the provider system to create Billing Entries in Provider Portal platform. It's an HTTP based API that different provider apps and system can use it programmatically to POST Billing Entries.

Using the Provider Upload API

HTTP/1.1

All data transfer conform to HTTP/1.1, and all endpoints require HTTPS.

Host URL

- Url for integrated testing environment - **https://provtest.ltssmaryland.org/**
- Url for production environment - TBD

Authentication

Authentication allows provider systems to create Billing Entry using Provider Upload API. Authentication also allow us to identify the provider system, the type of data being transmitted. Provider Upload API requires to have a JWT (JSON Web Token) each time you access the upload billing entry endpoint.

Authentication is the first request a system should make before begin their upload process.

Please request MDH to provide following provider specific credentials before using Provider Upload API

- AppId
- AppSecret
- SSO Username

Note: *SSO accounts for accessing provider upload API has different role and provider cannot use the same account for accessing Provider Portal and vice-versa.*

Authentication Endpoint

<url>/issue/oauth2/token

	Key	Value	Comments
Headers			
	Authorization	Basic {appId}:{appSecret}	appId and appSecret will be issued to each provider by MDH
	content-type	application/x-www-form-urlencoded	
Post Body : x-www-form-urlencoded			
	grant_type	password	Static value
	Username	{sso_username}	SSO Username will be provided by MDH
	password	{sso_password}	Providers will have the option reset their password.
	scope	{scopeurl}	Scope will be static value for all providers Scope url will be the same as UploadBillingEntry endpoint

Sample Authentication Request in cURL

```
curl -X POST \
  <url>/issue/oauth2/token \
  -H 'Authorization: Basic
UHJvdmlkZXJVcGxvYWRUZXN0Q2xpZW50SWQ6Vk95UCtUbFZobXJNWElJWnhBcmlEZnlZSDBqdE0zN
1pDdDJUZ09aM1Jidz0=' \
  -H 'Content-Type: application/x-www-form-urlencoded' \
  -H 'cache-control: no-cache' \
  -d 'grant_type=password&username=<username>&password=<password>&scope=<scopeurl>'
```

Note: The above sample request is for illustration purpose only. Every programming language will need to build the request in its native implementation.

Authentication Response

Success

	HTTP Response Code	200
Response Body - Sample		

```
{
        "access_token": "{TOKEN_STRING}",
        "token_type": "urn:ietf:params:oauth:token-type:jwt",
        "expires_in": {TOKEN_EXPIRATION_IN_SECONDS},
        "refresh_token": null
}
```

Fail		
Unauthorized	HTTP Response Code	401

Authentication Response Body Attributes

Response Body		
	access_token	**JWT token string. This token string needs to be used in every request for the upload process.**
	token_type	Specifies which token type is being returned. Token_type does not need to be used in subsequent calls. Token type would always be **urn:ietf:params:oauth:token-type:jwt**
	expires_in	Token expiration time in seconds. Default value is 28800 seconds (8 hours)
	refresh_token	Null. JWT token works on absolute expiration time and the token will expire after the specified time. Application needs to get another token if the token expires before the billing entry upload process.

Upload Billing Entry

After successful authentication, provider system can access provider upload web API resource to upload billing entries one at a time. A single billing entry can be uploaded at a time. The Web API is designed to keep the small pay load for better throughput.

Idempotent Design

Upload billing entry is idempotent by design. Each billing entry requires a unique transaction id. If for some reason the client experience an exception such as timeout, connection closed etc. Client system should use the same transaction Id for the retry call. This will avoid duplicate Billing Entry in provider portal system. You may reuse the same transaction Id when request fails due to validation error.

Upload Billing Entry Endpoint

<url>/Ltss.SelfServeApi.Web/api/providerupload/uploadbillingentry

Upload Billing Entry Request Data Format

Field	JSON Type	Required	Description
TransactionId	String	**Yes**	A unique transactionId for every request. A 128 bit UUID/GUID to avoid collision.
ServiceIdentifier	string	**Yes**	Alpha-numeric Abbreviated service names/codes titles will be provided by DDA.
ServiceDate	string	**Yes**	Valid Date of Service Not greater than current date Not older than 365 days Should be greater than client's pilot date (pilot phase only) mm-dd-yyyy format
ClientLtssId	string	**Yes**	Alpha-numeric Valid Client's LTSS ID
ProviderMa	string	**Yes**	Alpha-numeric Valid Provider MA Number
Units	number	**Yes**	Required if Cost is not specified Number of Units. A whole number No decimal is allowed
Cost	number	**Yes**	Required if Unit is not specified Cost of Activity up-to 2 decimal point

Data Contract for Upload Request

	Key	Value	Comments
Headers			
	Authorization	Bearer {TOKEN_STRING}	Use **access_token** string from the authentication

			response. This header is required for every upload request.
	content-type	application/json	
Post Body : application/json - Sample			

```
{
        "TransactionId": "00000000-0000-0000-0000-000000000000"
        "ServiceIdentifier": "xxxxxxxxxxxxxxx",
        "ServiceDate": "9/12/2018",
        "ClientLtssId": "A1234567890",
        "ProviderMaNumber": "A12345678",
        "Units": 2,
        "Cost": 5.4
}
```

Upload Billing Entry Response

Success			
	HTTP Response Code	200	Success
Response Header	TID	{TransactionId}	The same transaction Id from Request.
Fail			
Response Header	TID	{TransactionId}	The same transaction Id from Request.
	HTTP Response Code	401	Unauthorized This can be due to invalid or expired JWT.
		400	Bad request Data validation failed. 400 response will have ErrorCode and descriptive message for the validation that failed.
Response Body for HTTP Response code 400 : application/json – Sample			

```
{
        "Message": ""
        "ErrorCode":
}
```

Error Codes

The execution sequence of validations are in the order of Error codes. Provider upload api however will short circuit the validations and returns HTTP 400 error upon first failed validation.

Api at any point would always return a single error code and message.

Explanation of all Error codes and messages for HTTP response code 400		
Error Code	**Error Message**	**Validation Rule**
1000	Invalid SSO Username	No staff profile is associated with SSO username in LTSS
1100	TransactionId not present	Transaction Id is missing in request
1200	Invalid Date of Service	Date is not present in request. Date entered is invalid and cannot be parsed to a standard date. I.E. date is not in correct format. Date is less than dda-go-live (configured) date Difference between Service date and current date is more than 365 days.
1300	Invalid Unit or Cost	Cost and Units are absent in request. Or Both Cost and Units are specified and greater than 0 in request. Provided cost for unit-based service, or units for cost based service Unit should be a whole number Cost supports only upto 2 decimal point
1400	Duplicate TransactionId	Do not reuse TransactionId. An existing billing entry with the same transactionId will return error.
1500	Invalid Ltss Client Id	ClientId is blank in request ClientId does not exist in LTSS
1600	Invalid Provider MA#	Provider MA# is blank in request Provider MA# is not associated with any location
1700	Invalid Service Identifier	Service Identifier is blank in request Service Identifier is invalid
1800	Exceeds allowed cap limits	Cost specified is greater than max rate defined for the service group.

1900	Exceeds allowed cap limits	Units specified are greater than max units defined for the service group.